Package ‘apeglm’

March 20, 2024

Title Approximate posterior estimation for GLM coefficients
Version 1.24.0
Maintainer Anqi Zhu <anqizhu@live.unc.edu>
Description apeglm provides Bayesian shrinkage estimators for effect sizes for a variety of GLM models, using approximation of the posterior for individual coefficients.
VignetteBuilder knitr, rmarkdown
Imports emdbook, SummarizedExperiment, GenomicRanges, methods, stats, utils, Rcpp
Suggests DESeq2, airway, knitr, rmarkdown, testthat
LinkingTo Rcpp, RcppEigen, RcppNumerical
License GPL-2
Encoding UTF-8
LazyData true
biocViews ImmunoOncology, Sequencing, RNASeq, DifferentialExpression, GeneExpression, Bayesian
RoxygenNote 7.1.1
git_url https://git.bioconductor.org/packages/apeglm
git_branch RELEASE_3_18
git_last_commit b6b76e6
git_last_commit_date 2023-10-24
Repository Bioconductor 3.18
Date/Publication 2024-03-20
Author Anqi Zhu [aut, cre],
Joshua Zitovsky [ctb],
Joseph Ibrahim [aut],
Michael Love [aut]
apeglm

Approximate posterior estimation for GLM coefficients

Description

apeglm provides Bayesian shrinkage estimators for effect sizes in GLM models, using approximation of the posterior for individual coefficients.

Usage

apeglm(
  Y,
  x,
  log.lik,
  param = NULL,
  coef = NULL,
  mle = NULL,
  no.shrink = FALSE,
  interval.type = c("laplace", "HPD", "credible"),
  interval.level = 0.95,
  threshold = NULL,
  contrasts,
  weights = NULL,
  offset = NULL,
  flip.sign = TRUE,
  prior.control,
  multiplier = 1,
  ngrid = 50,
  nsd = 5,
  ngrid nuis = 5,
  nsd nuis = 2,
  log.link = TRUE,
  param.sd = NULL,
  method = c("general", "nbinomR", "nbinomCR", "nbinomC", "nbinomC*", "betabinR",
            "betabinCR", "betabinC", "betabinC*")
  optim.method = "BFGS",
  bounds = c(-Inf, Inf)
)
Arguments

Y the observations, which can be a matrix or SummarizedExperiment, with columns for samples and rows for "features" (e.g. genes in a genomic context). If Y is a SummarizedExperiment, apeglm will return, in addition to other list items, a GRanges or GRangesList ranges with the estimated coefficients as metadata columns.

x design matrix, with intercept in the first column. Continuous-valued columns should be centered and scaled to unit variance, or at least set so that the scale (in SD) is not very large or very small.

log.lik the log of likelihood function, specified by the user. For Negative Binomial distribution, user can use logLikNB provided within the package.

param the other parameter(s) to be used in the likelihood function, e.g. the dispersion parameter for a negative binomial distribution. this can be a vector or a matrix (with columns as parameters)

coef (optional) the index of the coefficient for which to generate posterior estimates (FSR, svalue, and intervals)

mle (optional) a 2 column matrix giving the MLE and its standard error of coef. this will be used to adapt the scale of the prior (empirical Bayes). This overrides the prior.scale specified by prior.control and sets no.shrink to all coefficients other than coef. Note that these MLE’s and SE’s should be on the natural log scale for a log link GLM.

no.shrink logical, if TRUE, apeglm won’t perform shrinkage (default is FALSE)

interval.type (optional) can be "laplace", "HPD", or "credible", which specifies the type of Bayesian interval that the user wants to output; "laplace" represents the Laplace approximation of the posterior mode

interval.level (optional) default is 0.95

threshold (optional) a threshold for integrating posterior probabilities, see details under 'Value'. Note that this should be on the natural log scale for a log link GLM.

contrasts (optional) contrast matrix, same number of rows as x

weights (optional) weights matrix, same shape as Y

offset (optional) offsets matrix, same shape as Y. Note that this should be on the natural log scale for a log link GLM.

flip.sign whether to flip the sign of threshold value when MAP is negative, default is TRUE (threshold must then be positive)

prior.control see Details

multiplier a positive number, when the prior is adapted to the mle matrix provided, this parameter connects the scale of the estimated distribution of betas to the scale of the prior. the default value was chosen based on FSR and error analysis of simulated data

ngrid the number of grid points for grid integration of intervals

nsd the number of SDs of the Laplace posterior approximation to set the left and right edges of the grid around the MAP

ngrid.nuis the number of grid points for nuisance parameters
apeglm

nsd.nuis  the number of Laplace standard errors to set the left and right edges of the grid around the MAP of the nuisance parameters

log.link  whether the GLM has a log link (default = TRUE)

param.sd  (optional) potential uncertainty measure on the parameter param. this should only be a vector, used when param specifies a single parameter

method  options for how apeglm will find the posterior mode and SD. The default is "general" which allows the user to specify a likelihood in a general way. Alternatives for faster performance with the Negative Binomial likelihood are: "nbinomR", "nbinomCR", and "nbinomC" / "nbinomC*". These alternative methods should provide increasing speeds, respectively. (Also for beta binomial, substitute "betabin" for "nbinom" in the above.) From testing on RNA-seq data, the nbinom methods are roughly 5x, 10x and 50x faster than "general". Note that "nbinomC" uses C++ to find the MAP for the coefficients, but does not calculate or return the posterior SD or other quantities. "nbinomC*" is the same as "nbinomC", but includes a random start for finding the MAP. "nbinomCR" uses C++ to calculate the MAP and then estimates the posterior SD in R, with the exception that if the MAP from C++ did not converge or gives negative estimates of posterior variance, then this row is refit using optimization in R. These alternatives require the degrees of freedom for the prior distribution to be 1, and will ignore any function provided to the log.lik argument. param should specify the dispersion parameter of a Negative Binomial (such that Var = mu + param mu^2).

optim.method  the method passed to optim

bounds  the bounds for the numeric optimization

Details

to a Normal prior on the intercept, and a t prior on the non-intercept coefficients (similar to bayesglm in the arm package. The following are defaults:

• no.shrink = 1: index of the coefficient(s) not to shrink
• prior.mean = 0: mean of t prior
• prior.scale = 1: scale of t prior
• prior.df = 1: df of t prior
• prior.no.shrink.mean = 0: mean of Normal
• prior.no.shrink.scale = 15: scale of Normal

So without specifying prior.control, the following is set inside apeglm:

prior.control <- list(no.shrink=1,prior.mean=0,prior.scale=1,prior.df=1,prior.no.shrink.mean=0,prior.no.shrink.scale=15)

Note that the prior should be defined on the natural log scale for a log link GLM.
Value

a list of matrices containing the following components:

- map: matrix of MAP estimates, columns for coefficients and rows for features
- sd: matrix of posterior SD, same shape as map
- prior.control: list with details on the prior
- fsrc: vector of the false sign rate for coef
- svalue: vector of the s-values for coef
- interval: matrix of either HPD or credible interval for coef
- thresh: vector of the posterior probability that the estimated parameter is smaller than the threshold value specified in threshold when MAP is positive (or greater than -1 * threshold value when MAP is negative and flip.sign is TRUE)
- diag: matrix of diagnostics
- contrast.map: vector of MAP estimates corresponding to the contrast when contrast is given
- contrast.sd: vector of posterior SD corresponding to the contrast when contrast is given
- ranges: GRanges or GRangesList with the estimated coefficients, if Y was a SummarizedExperiment.

Note that all parameters associated with coefficients, e.g. map, sd, etc., are returned on the natural log scale for a log link GLM.

References

Adaptive Cauchy prior:

False sign rate and s-value:

Examples

```r
# Simulate RNA-Seq read counts data

# 5 samples for each of the two groups
# a total of 100 genes
n.per.group <- 5
n <- n.per.group * 2
m <- 100

# The design matrix includes one column of intercept
# and one column indicating samples that belong to the second group
ccondition <- factor(rep(letters[1:2], each = n.per.group))
x <- model.matrix(~condition)
```
# Specify the standard deviation of beta (LFC between groups)
beta.sd <- 2
beta.cond <- rnorm(m, 0, beta.sd)
beta.intercept <- runif(m, 2, 6)
beta.mat <- cbind(beta.intercept, beta.cond)

# Generate the read counts
mu <- exp(t(x %*% t(beta.mat)))
Y <- matrix(rnbinom(m*n, mu=mu, size=1/.1), ncol = n)

# Here we will use the negative binomial log likelihood
# which is an exported function. See 'logLikNB' for details.
# For the NB:
# 'param' is the dispersion estimate (1/size)
# 'offset' can be used to adjust for size factors (log of size factors)
param <- matrix(0.1, nrow = m, ncol = 1)
offset <- matrix(0, nrow = m, ncol = n)

# Shrinkage estimator of betas:
# (for adaptive shrinkage, 'apeglm' requires 'mle' coefficients
# estimated with another software, or by first running 'apeglm'
# setting 'no.shrink=TRUE'.)
res <- apeglm(Y = Y, x = x,
log.lik = logLikNB,
param = param,
offset = offset,
coef = 2)

head(res$map)
plot(beta.mat[,2], res$map[,2])
abline(0,1)

---

**bbEstDisp**

*Simple line-search estimator for dispersion of a beta binomial*

**Description**

Uses R’s optimize function to find the maximum likelihood estimate of dispersion for a beta binomial distribution (theta for the dbetabinom function in the emdbook package). The counts, size, and beta are matrices, such that each row could be treated as a beta-binomial GLM problem.

**Usage**

`bbEstDisp(success, size, weights = 1, x, beta, minDisp, maxDisp, se = FALSE)`

**Arguments**

- `success`: the observed successes (a matrix)
- `size`: the total trials (a matrix)
weights the weights (1 or a matrix)
x the design matrix, as many rows as columns of success and size
beta a matrix of MLE coefficients, as many rows as success and size
minDisp the minimum dispersion value
maxDisp the maximum dispersion value
se logical, whether to return standard error estimate on the log of the dispersion (theta). Warning: the standard error estimates are not reliable at the boundary (log of minDisp and maxDisp), and should be interpreted with caution!

Value

a vector of estimated dispersions (theta). if se=TRUE a matrix with columns: the vector of estimated dispersions and the standard errors for the log of the estimated dispersions

Examples

library(emdbook)
n <- 100
m <- 100
size <- matrix(rnbinom(n*m, mu=100, size=10),ncol=m)
success <- matrix(rbetabinom(n*m, prob=.5, size=size, theta=100),ncol=m)
x <- matrix(rep(1,m),ncol=1)
beta <- matrix(rep(0,n),ncol=1)
theta <- bbEstDisp(success=success, size=size, x=x, beta=beta, minDisp=1, maxDisp=500)
summary(theta)

# with standard error estimates on log of dispersion
fit <- bbEstDisp(success=success, size=size, x=x, beta=beta, minDisp=1, maxDisp=500, se=TRUE)
plot(fit[1:20,"theta"], ylim=c(0,500), ylab="theta-hat")
log.theta <- log(fit[1:20,"theta")
log.theta.se <- fit[1:20,"se"]
segments(1:20, exp(log.theta - 2 * log.theta.se),
1:20, exp(log.theta + 2 * log.theta.se))
abline(h=100,col="red")

logLikNB

Log likelihood for Negative Binomial

Description

This is a simple function to be passed to apeglm as a log likelihood for the negative binomial distribution.

Usage

logLikNB(y, x, beta, param, offset)
Arguments

- **y**  
  the counts

- **x**  
  a design matrix

- **beta**  
  the coefficient vector (natural log scale)

- **param**  
  the overdispersion (alpha in DESeq2 notation)

- **offset**  
  an offset matrix (natural log scale)

Value

the log likelihood for each sample in y

Examples

```r
# this function is used by 'apeglm' to specify  
# a negative binomial log likelihood.  
# so it's only passed as an argument, not for use on its own.  
# we can show its output nevertheless:

y <- rnbinom(10, mu=100, size=1/.1)
x <- cbind(rep(1,10),rep(0:1,each=5))
beta <- c(log(100),0)
param <- .1
offset <- rep(0, 10)
logLikNB(y, x, beta, param, offset)
```

---

### svalue

Local FSR to svalue

Description

Convert local FSR to svalue by taking the cumulative mean of increasing local FSRs. This is used within apeglm to generate the svalue table, but provided as convenience function.

Usage

```
svalue(lfsr)
```

Arguments

- **lfsr**  
  the local false sign rates

Value

s-values
References

False sign rate and s-value:


Examples

# Example 1 -- simulated local FSR data
local.fsr <- runif(1000)
sval <- svalue(local.fsr)

# Example 2 -- first runs example from 'apeglm'
example("apeglm")
local.fsr <- res$fdr[1,]
sval <- svalue(local.fsr)
Index

apeglm, 2
bbEstDisp, 6
logLikNB, 7
svalue, 8